Ogden Air Logistics Center

U.S. AIR FORCE

Reliability Index for the USMC Expeditionary Fighting Vehicle

SSTC 2008 – 1 May 2008

Robert J. Knapper
David L. Jolley
David R. Webb

Outline

- Expeditionary Fighting Vehicle
- Traditional Reliability Measures
- Why Software is Different
- Full-Lifecycle Software Reliability Measurement
- 309th SMXG
- Software Reliability Issues
- TSP Process Quality Index
- Software Reliability Index
- Pros and Cons
- Conclusions
- Contact Information

Expeditionary Fighting Vehicle
Overview

- **USMC Expeditionary Fighting Vehicle (EFV)**
  - Armored amphibious vehicle capable of seamlessly transporting Marines from Naval ships located beyond the visual horizon to inland objectives
  - Primary means of tactical mobility for the Marine Rifle Squad during the conduct of amphibious operations and subsequent ground combat operations ashore.
  - Keystone for both the Marine Corps Expeditionary Maneuver Warfare and Ship-to-Objective Maneuver warfighting concepts
- **EFV PMO – Program Manager, Advanced Amphibious Assault**
- **Prime Contractor – General Dynamics Amphibious Systems**
- **Status**
  - **Currently in the System Development and Demonstration (SDD) Acquisition Phase**
  - **Operational Assessment in FY 2011**
  - **Low-Rate Initial Production Scheduled for FY 2012**

Expeditionary Fighting Vehicle
Mission Capabilities

OGDEN AIR LOGISTICS CENTER


Move - Land


Carry


Move - Water


Shoot


Communicate

Expeditionary Fighting Vehicle Characteristics

OGDEN AIR LOGISTICS CENTER

- General
  - Amphibious Tracked Vehicle
  - “Drive-by-wire”
  - Land mobility at 45mph on par with M1 Abrams Tank
  - Water mobility from over the horizon to shore at 25kts
  - Armed and armored to engage and protect
  - Net-ready
- Personnel Variant
  - Carries Marine Rifle Squad for conduct of amphibious operations and subsequent ground combat operations ashore
  - Incorporates MK46 Weapon System
    - MK44 30mm High Velocity Cannon and 7.62mm Coax Machine Gun
    - Fully Stabilized with Digital Fire Control and Thermal Sight for all-weather / all-night lethality
- Command Variant
  - Carries C2 systems and Operators
  - Provides Situational Awareness at the Squad Level
  - Provides Command and Control at the Battalion/Regimental Level.

Expeditionary Fighting Vehicle
Software

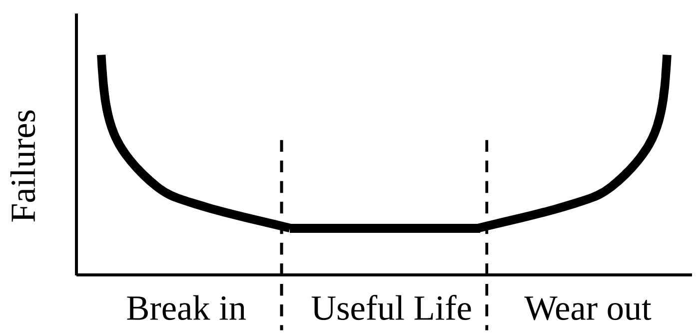
- Software Control
  - Vehicle Electronics Systems for Mobility, Fire Control, and C2 supported by real-time, embedded software
- GDAMS developed software for first set of SDD EFV prototypes (9 P-Variant, 1 C-Variant)
  - SDD-1 software used for first OA and Design for Reliability Effort following OA
  - Provided valuable feedback for verifying and validating SDD-2 software requirements
- Long-term Program strategy shifted future software development and maintenance to a Government activity
  - 309 SMXG selected for SDD-2 development and follow-on sustainment activity
- Both short-term and long-term reliability measures for 309 SMXG generated SDD-2 software is needed

Traditional Reliability Measures

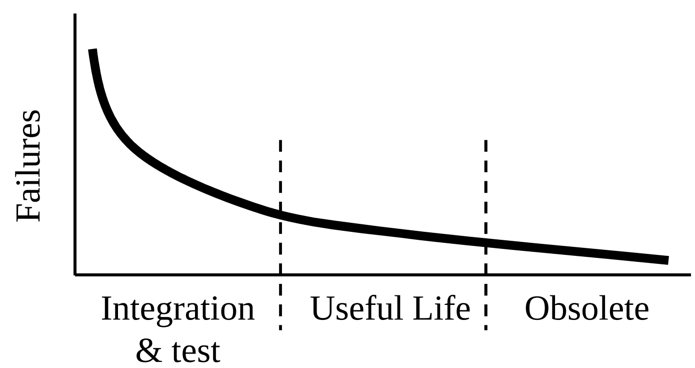
- Operational Definition of Reliability:
  - Mean Time to Failure (MTTF) combined with Mean Time to Repair (MTTR) is essentially an availability measure
- Focus primarily on defect identification and removal
  - E.g. Rayleigh Model looks at defect density rates over time as well as cumulative defect arrival patterns
  - Incorporated into lifecycle modeling for post-deployment
    - Maintenance cycles
    - Spares
- Equate Quality with Reliability
  - Fewer defects means higher reliability
- Availability Improvement
  - Mitigate MTTF/MTTR risks with maintenance scheduling and execution tailored to expected failure rates

Why Software is Different

- Traditional Mean Time to Failure metric does not adequately apply to software
- Software does not "wear out" like hardware
  - Mechanical and electronic components weaken with age.
  - Software does not weaken with age, but over time may become OBE or simply obsolete.



Hardware Failure Rate



Software Failure Rate

Full-Lifecycle Software Reliability Measurement

- Common Software Reliability Measurement
  - Entire Development Phase
  - Testing Phase
- Promoting Reliability Improvement
  - Front-end and back-end phases of the lifecycle are key contributors
- Need to Measure for Reliability at all phases
  - Requirements
  - Architecture and Design
  - Coding
  - Integration and Test
  - Deployment
  - Sustainment
- 309 SMXG is Piloting a Full-Lifecycle Software Reliability Measurement activity for EFV

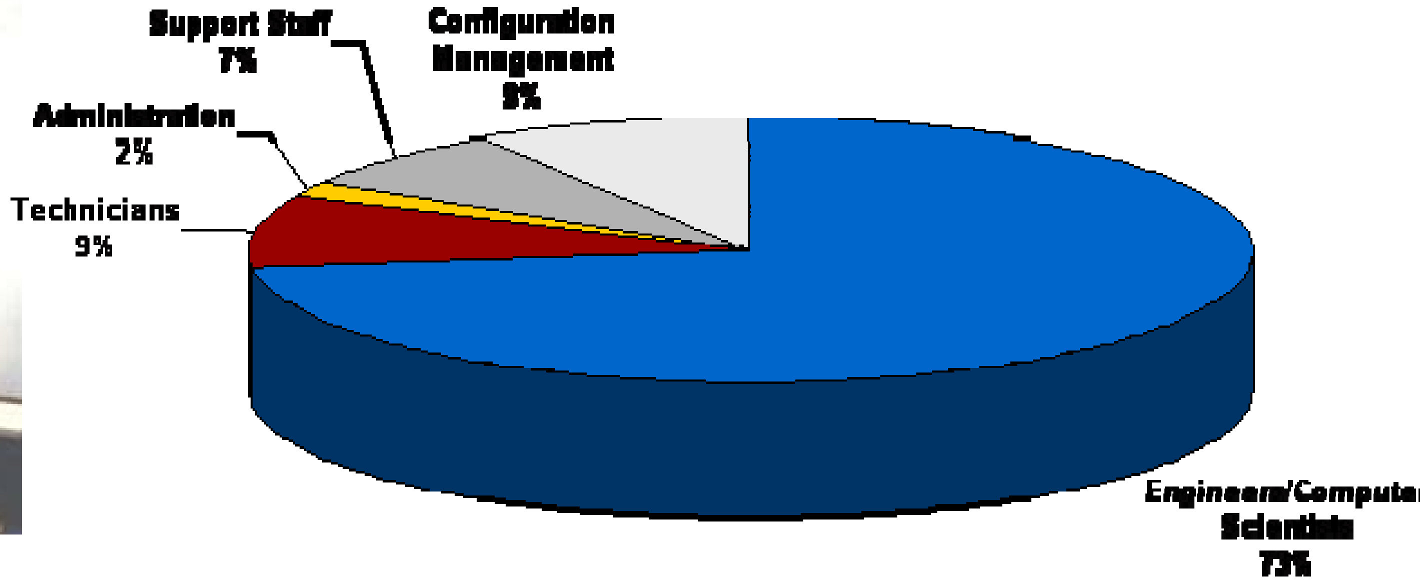
309th SMXG
Hill AFB - Utah

OGDEN AIR LOGISTICS CENTER

309th SMXG
Resources

Large Cadre of Talented People:

- 800+ Personnel
- Average over 10 years technical experience
- Growing by ~50 PEs/Year

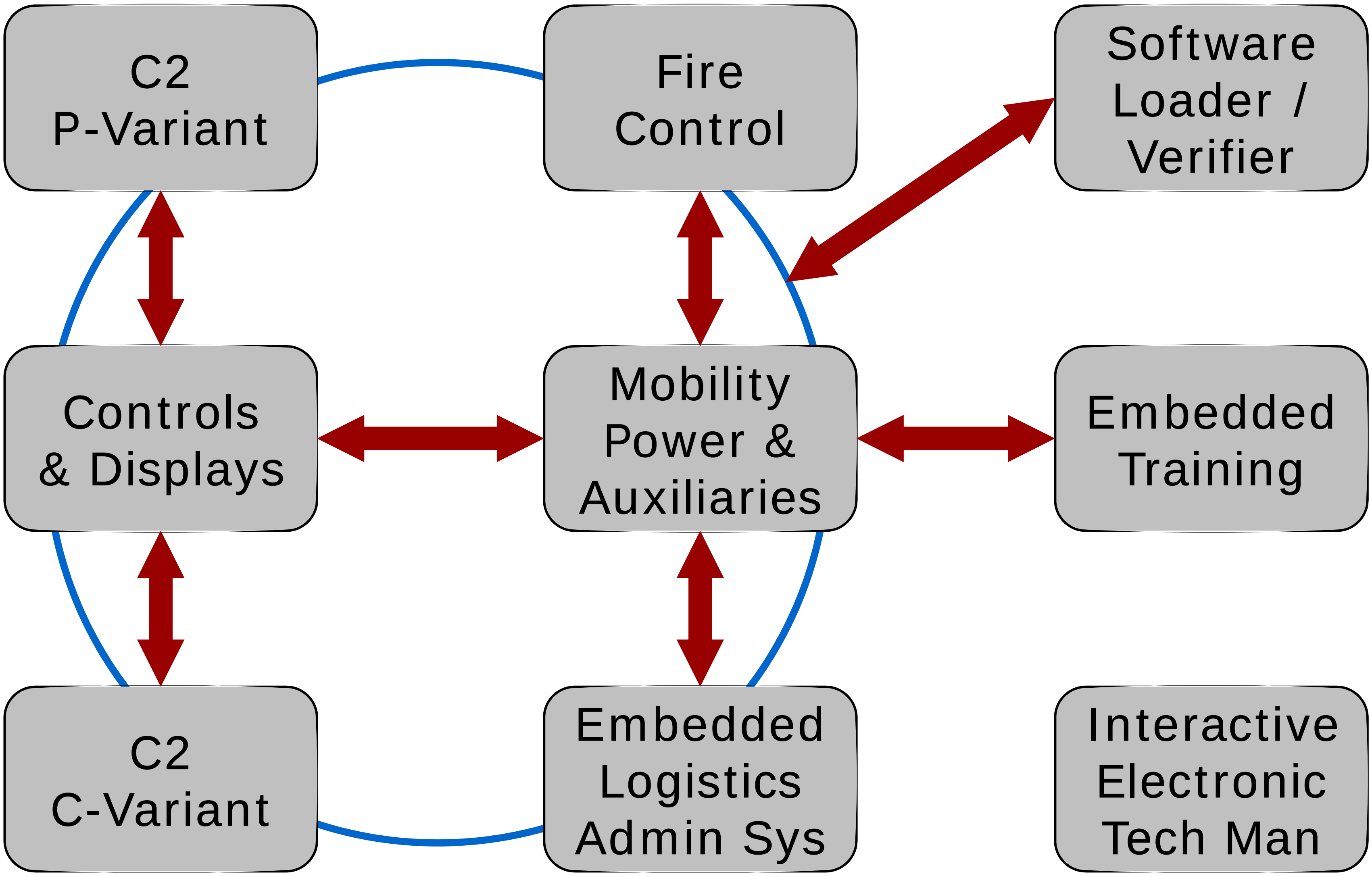


309th SMXG
Process Improvement Leader

OGDEN AIR LOGISTICS CENTER

- Focused on process improvement since 1991
- Assessed in 1998 to be Capability Maturity Model (CMM) - Level 5
  - **The highest rated level possible**
  - **First DoD government organization to receive CMM Level 5 rating**
- Earned AS9100 & ISO 9001 Registration in 2006
- Assessed in 2006 to be Capability Maturity Model Integration (CMMI) – Level 5
  - **Ranks SMXG in top 4% of all organic software organizations**
  - **Only government organization continuously rated CMM/CMMI level 5 since 1998**

309th SMXG
EFV Software Configured Items



309th SMXG
Traditional S/W Quality Measures

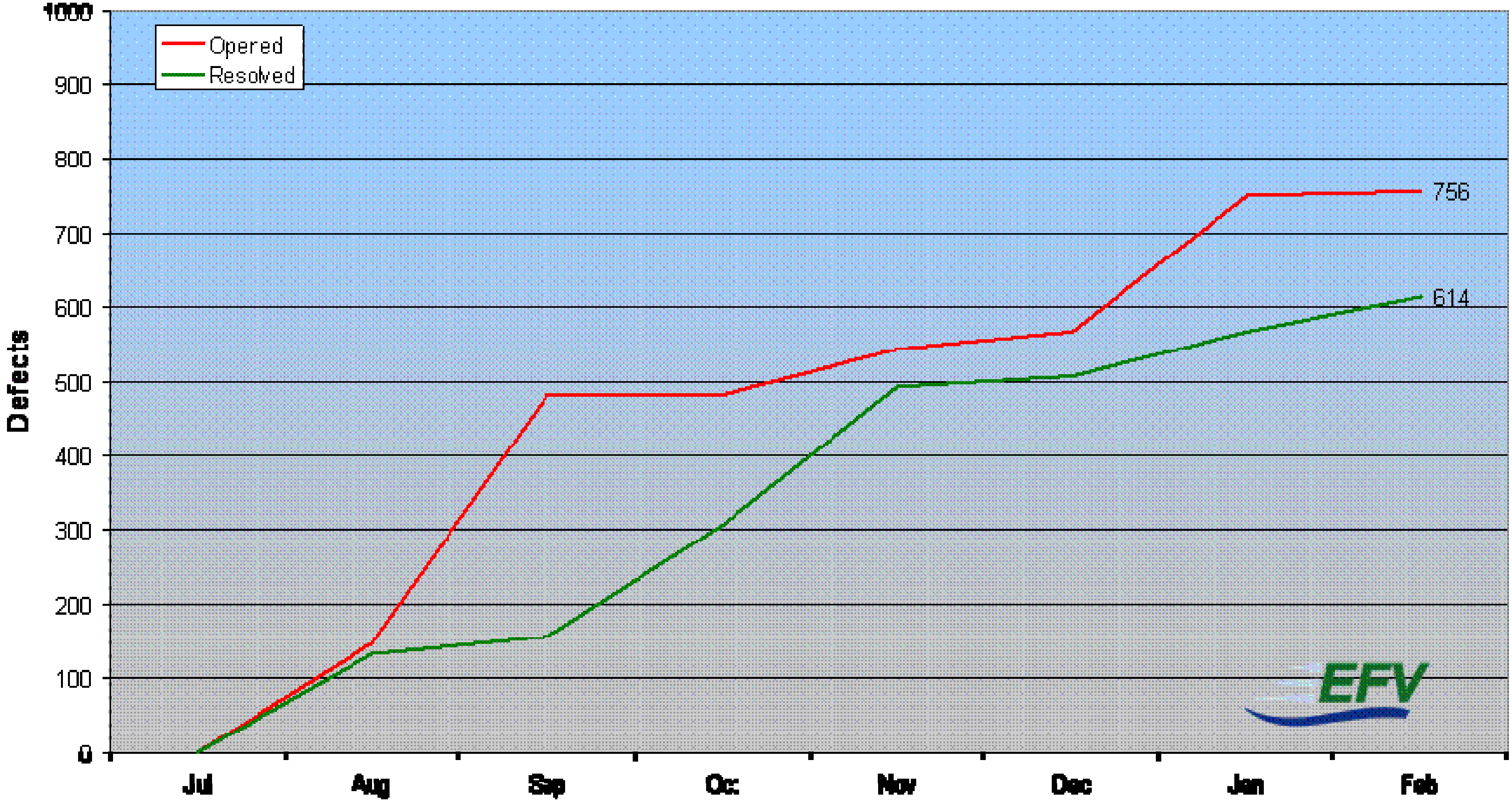
OGDEN AIR LOGISTICS CENTER

- **Product**
  - **Defects Injected** — Number of defects introduced within a phase
  - **Defects Detected** — Number of defects found within a phase
  - **Defects Removed** — Number of defects removed during a phase

- **Process**
  - **Defect Injection Rate** — Defects injected per hour per phase
    - GOAL: **< 0.4 per hour per phase**
    - 309th Average: **0.03 per 1000 Hours**
  - **Defect Detection Ratio** — Percent of all defects found internally
    - GOAL: **100% at System Test**
    - 309th Average: **96%**
  - **Defect Density** — Defects injected per build
    - GOAL: **Zero Defects at System Test**
    - 309th Average: **0.2 Defects per 1000 SLOC**
  - **Percent Rework** — Percent of total effort required to fix defects
    - GOAL: **< 10%**
    - 309th Average: **2%**

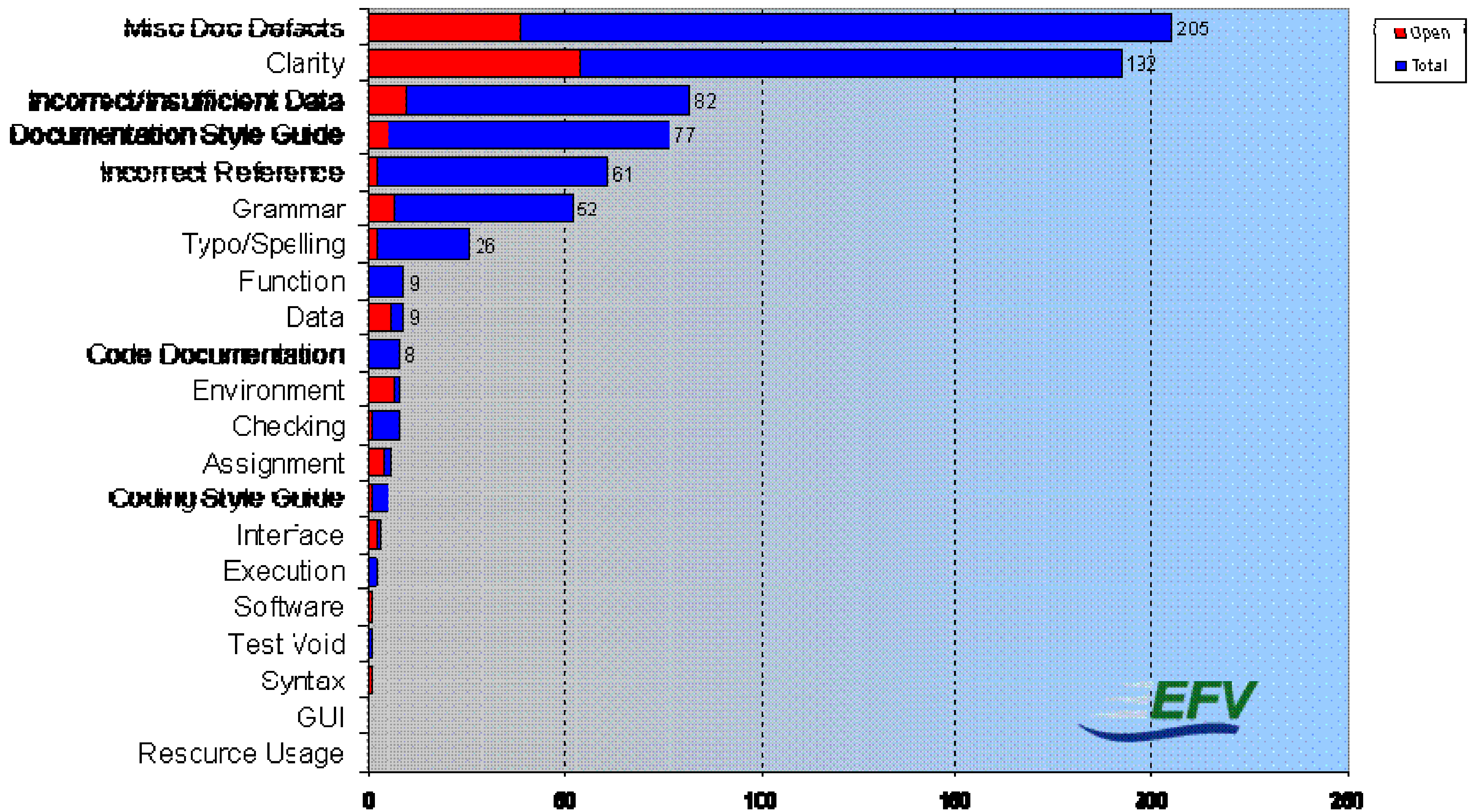
309th SMXG
Tracking Total Defects

OGDEN AIR LOGISTICS CENTER



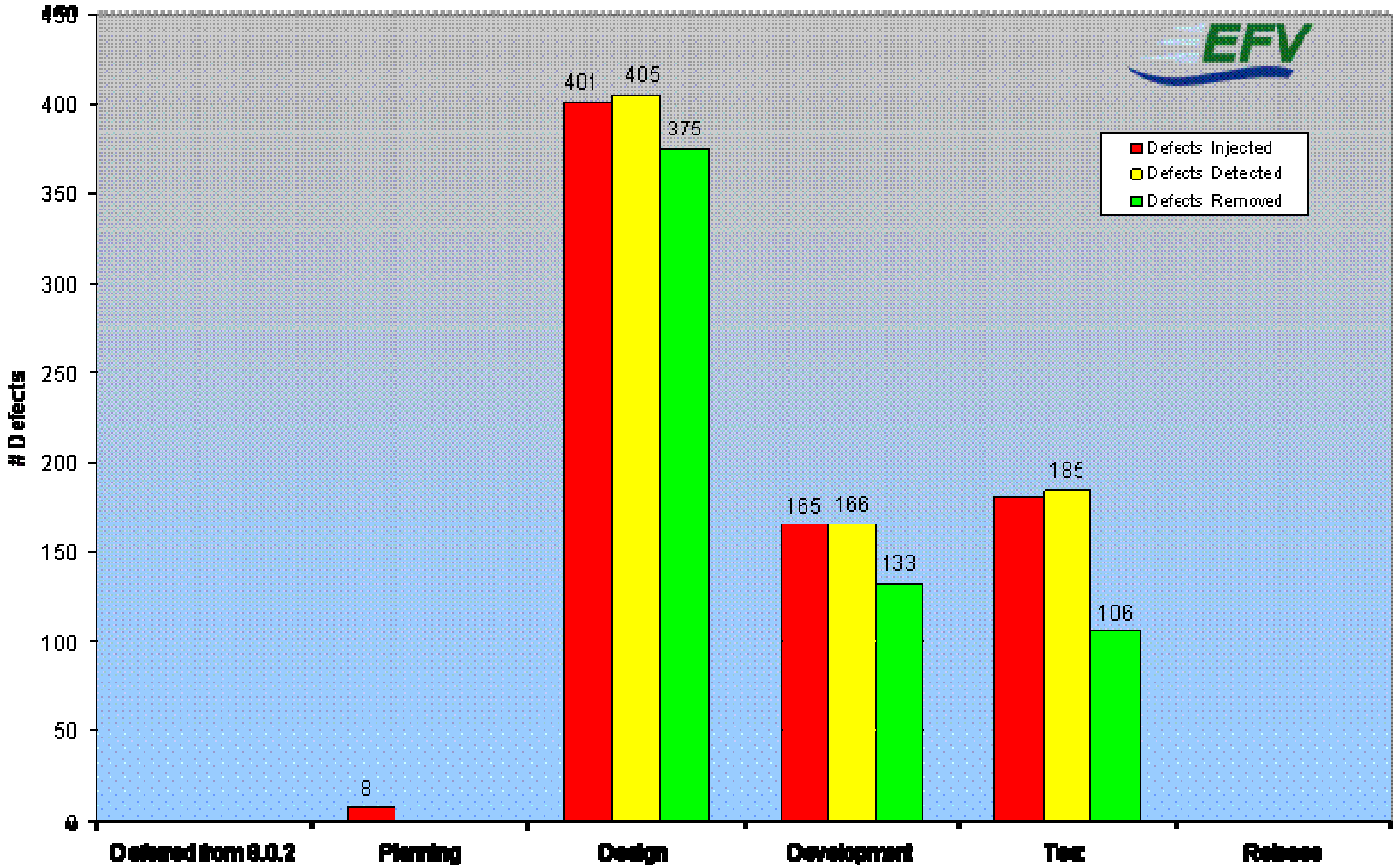
309th SMXG
Tracking Defect Types

OGDEN AIR LOGISTICS CENTER



309th SMXG
Tracking Product Quality

OGDEN AIR LOGISTICS CENTER



Software Reliability

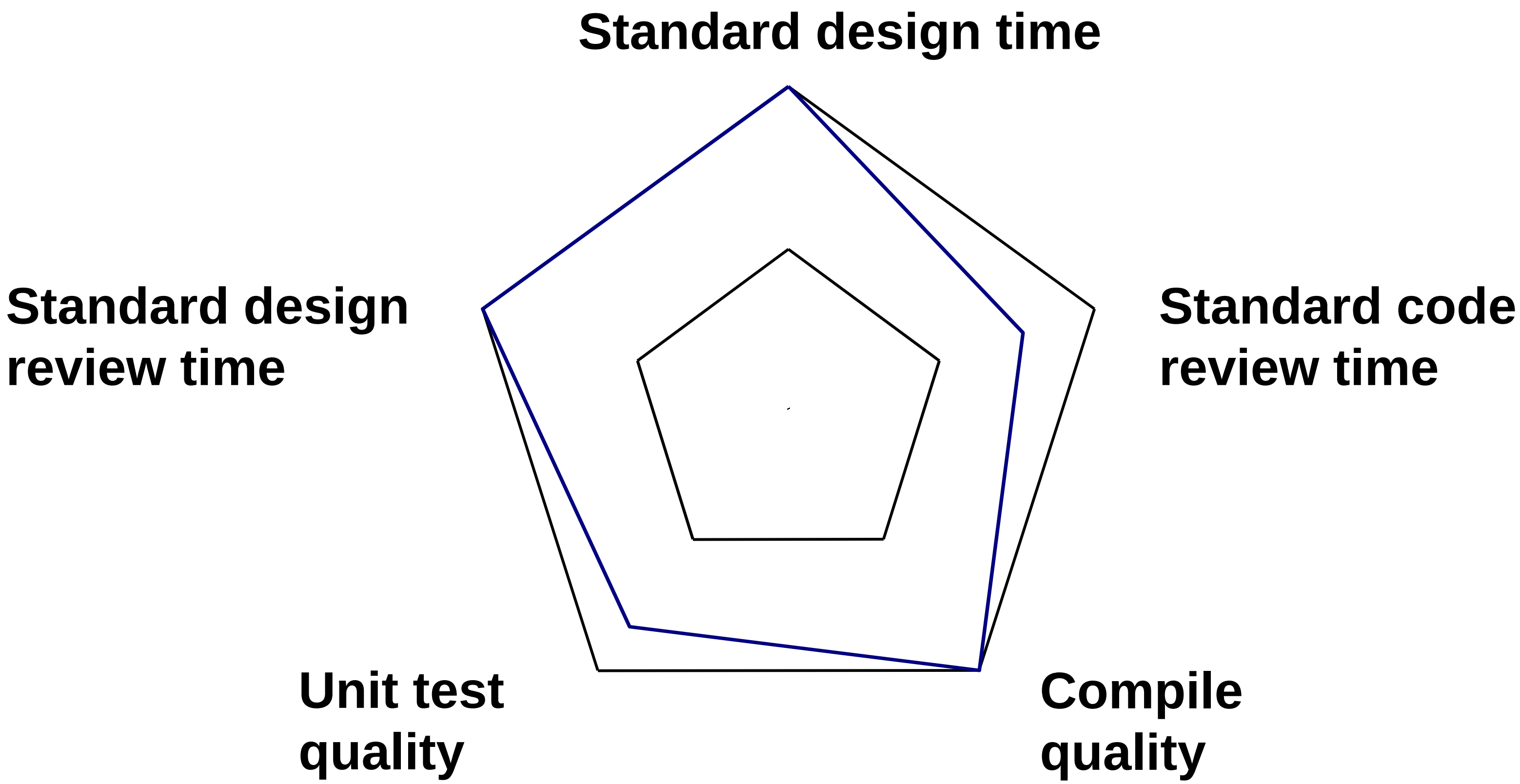
- Software Reliability Issues
  - How do we measure / control the process “up stream?”
  - What measures do we take?
    - Process Quality
    - Product Quality
    - The “ilities”
      - Install-ability
      - Usability
      - Flexibility
      - Maintainability
      - Portability
      - Etc.
  - While much has been written about this, there is no common way of evaluating software reliability other than tracking defect rates
  - It is difficult to track software reliability issues and make mid-course corrections

The TSP^{SM} Process Quality Index – 1

- The Software Engineering Institute has developed a tool for use the Team Software Process^{SM} teams to measure quality
- Teams measure the quality of the *process* used to produce their software in terms of *measurable goals*
- A high quality *process* typically produces a high quality *product*
- The TSP Process Quality Index (PQI) is a product of five factors:
  - Design Time (Goal: Design Time $\geq$ Code Time)
  - Review Time (Goal: Review Time $\geq$ 50% of Phase Time)
  - Compile Defect Density (Goal: $\leq$ 10 defects/KLOC)
  - Unit Test Defect Density (Goal: $\leq$ 5 defects/KLOC)
- TSP PQI goals are adjusted so that “1” represents meeting the goal

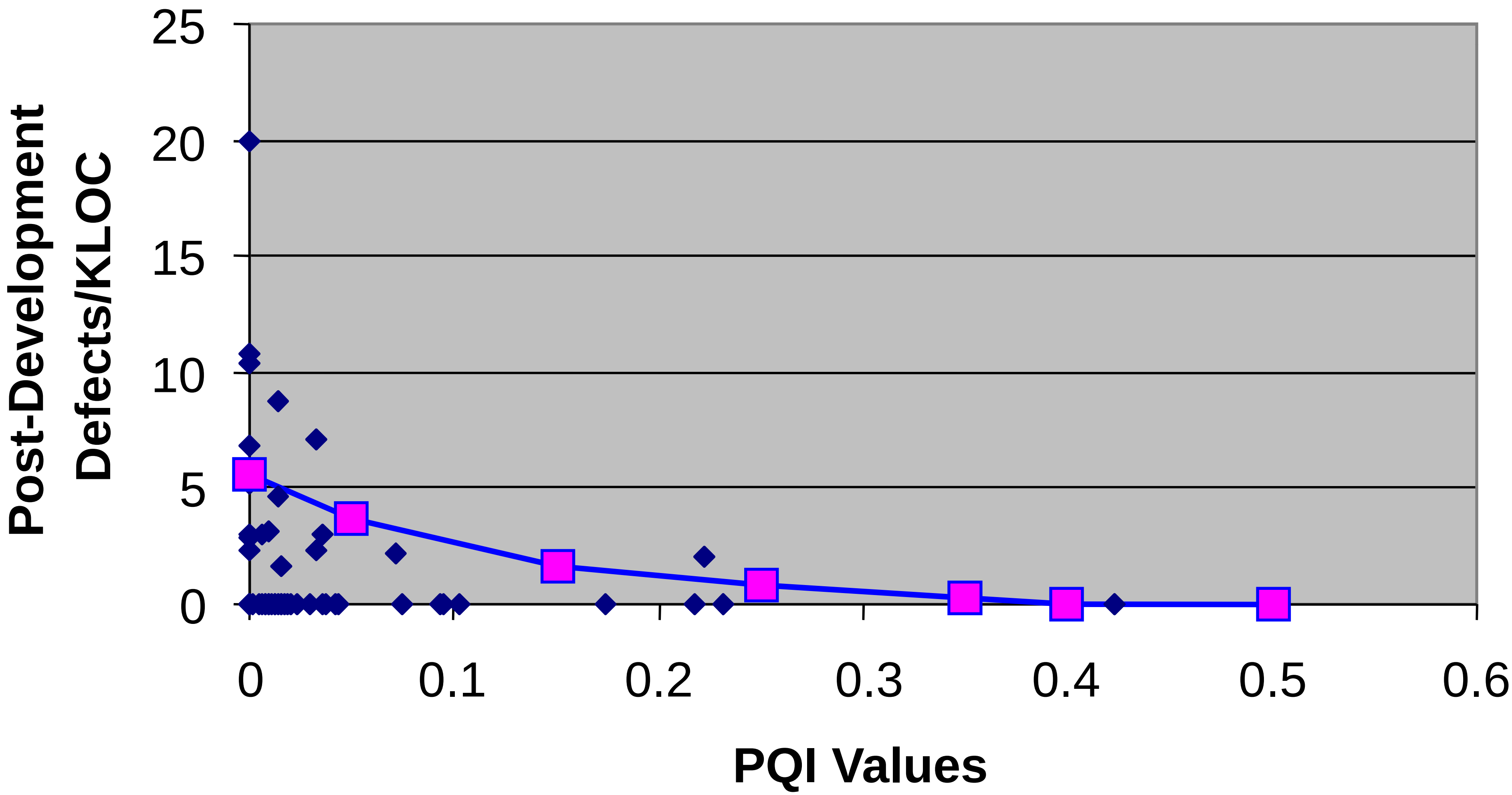
^{SM}Team Software Process and TSP are Service Marks of Carnegie Mellon University

The TSP PQI – 2



Source: Software Engineering Institute's "Managing TSP Teams" © 2006 by Carnegie Mellon University

PQI vs. Post-development Defects

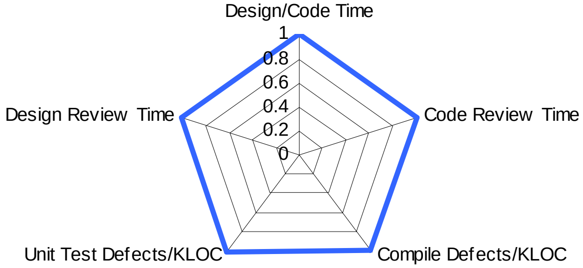


Source: Software Engineering Institute's "Managing TSP Teams" © 2006 by Carnegie Mellon University

Selected TSP Quality Profiles

Quality Profile for Assembly 1

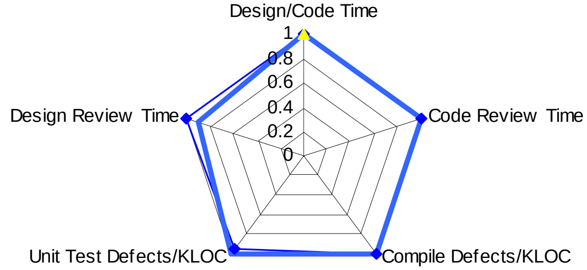
Test defects = 0



PQI = 0.97

Quality Profile for Assembly 2

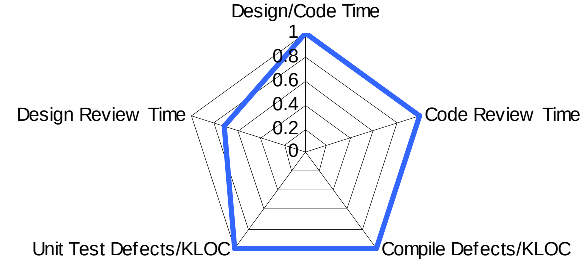
Test defects = 0



PQI = 0.88

Quality Profile for Assembly 3

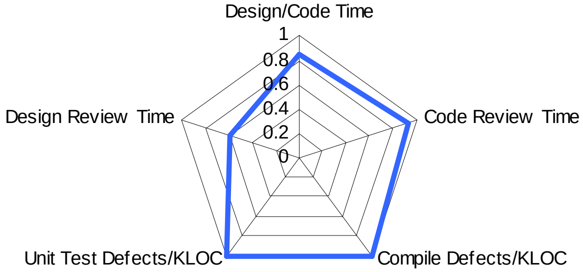
Test defects = 0



PQI = 0.71

Quality Profile for Assembly 4

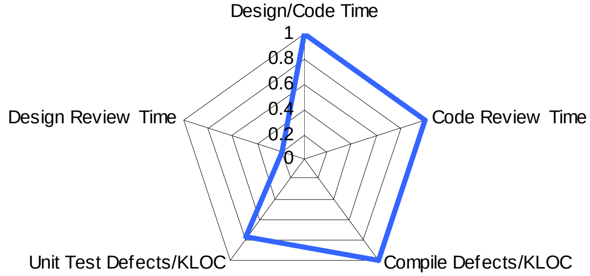
Test defects = 0



PQI = 0.59

Quality Profile for Assembly 5

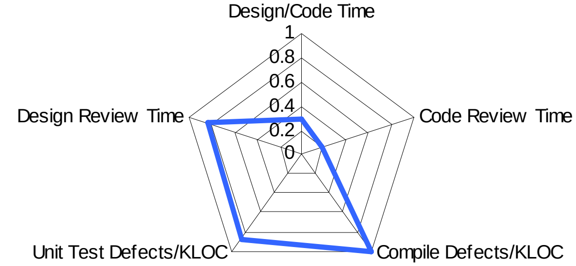
Test defects = 1



PQI = 0.15

Quality Profile for Assembly 6

Test defects = 3



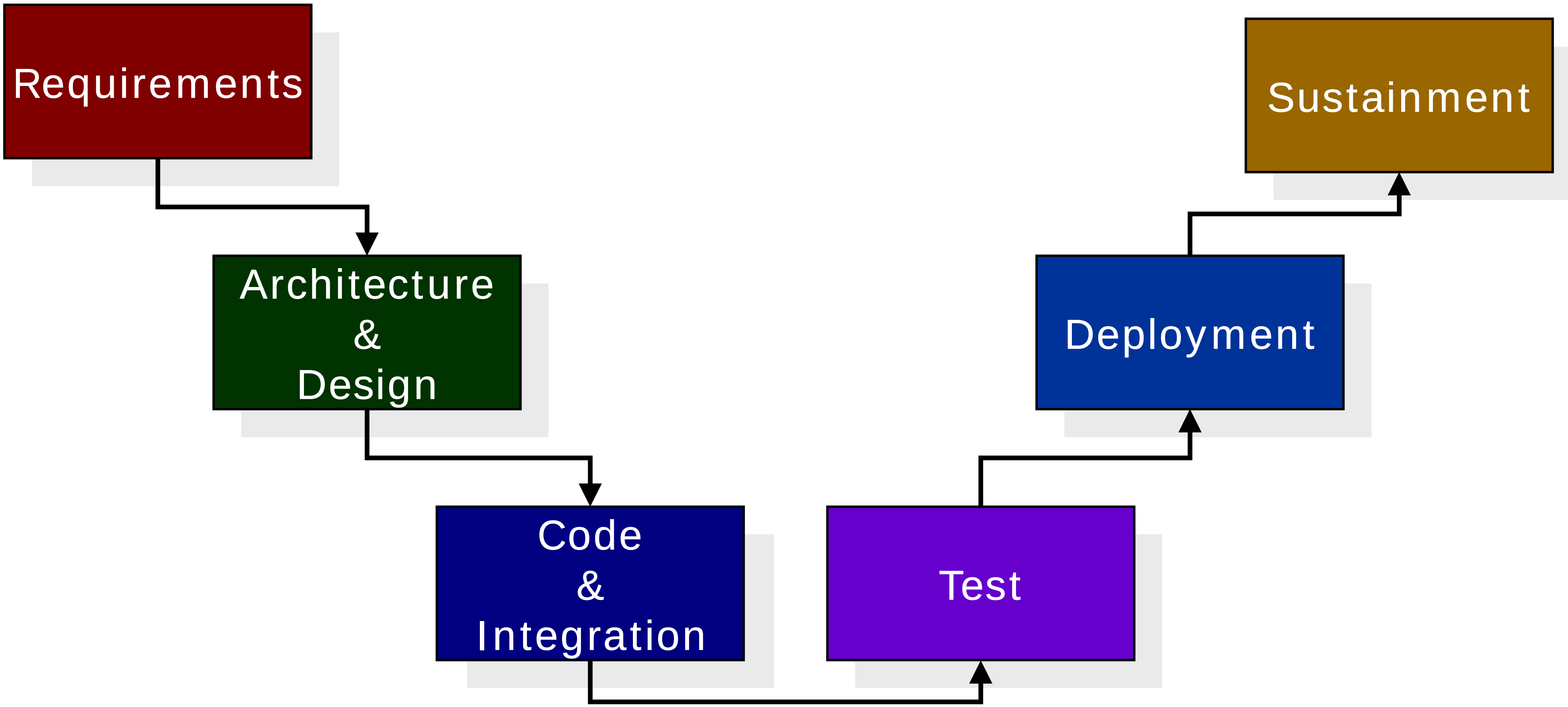
PQI = 0.04

Software Reliability Index (SRI)

- **The SRI Must:**
  - **Be measurable throughout the software lifecycle**
  - **Be independent of the software lifecycle selected (waterfall, spiral, iterative, etc.)**
  - **Provide early warning signs of reliability issues**
  - **Provide ability to make effective mid-course corrections**
- **SRI Format:**
  - **A format similar to TSP PQI would be helpful**
  - **As in TSP PQI, a set of measurable objectives that promote high reliability would be required**
  - **Measures must be adjusted to a number between 0 (unreliable) and 1 (highly reliable)**
  - **Measures must directly point to possible reliability issues**
  - **One approach is to use measurements from each software lifecycle phase**

Software Lifecycle
Typical Phases

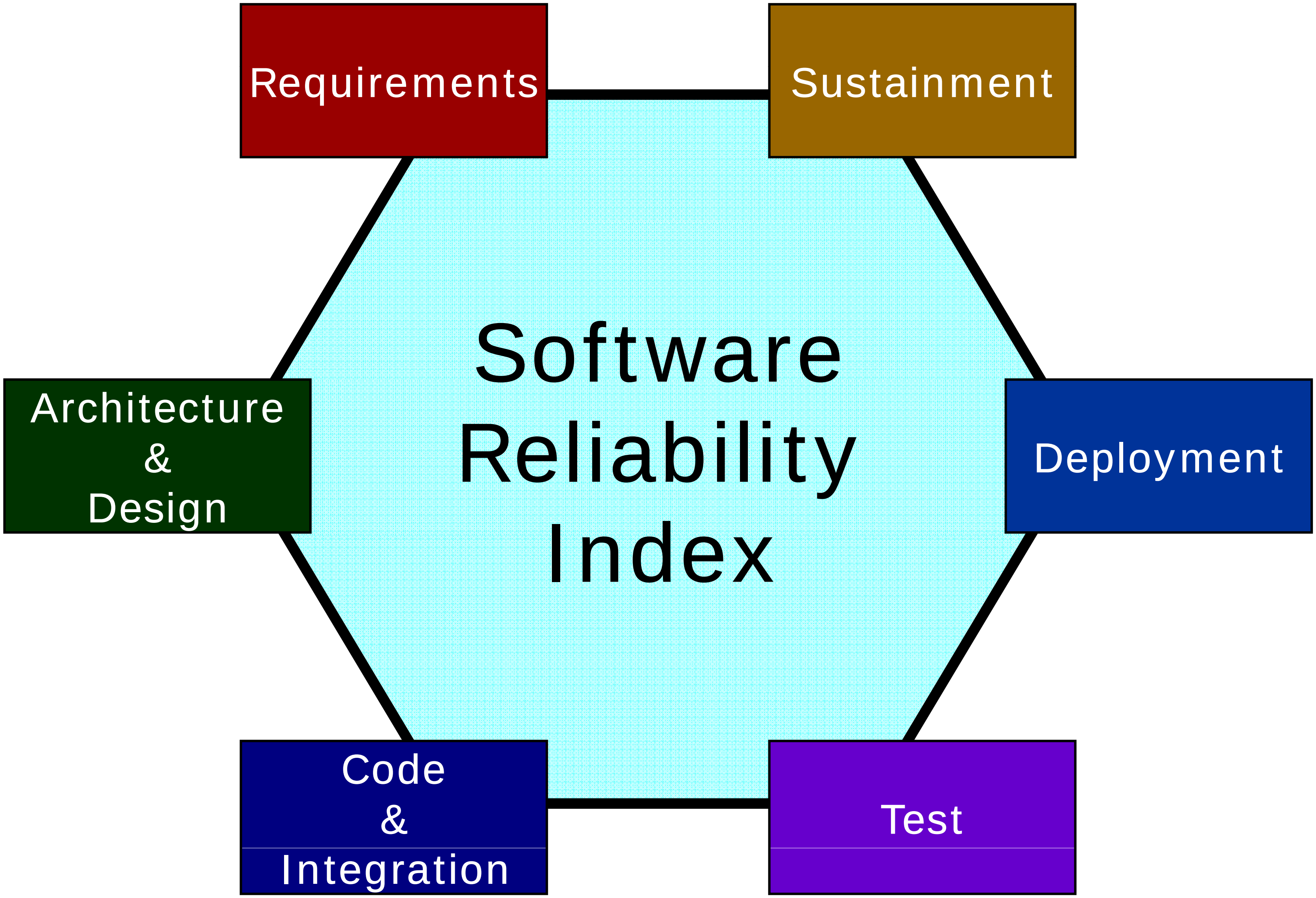
OGDEN AIR LOGISTICS CENTER



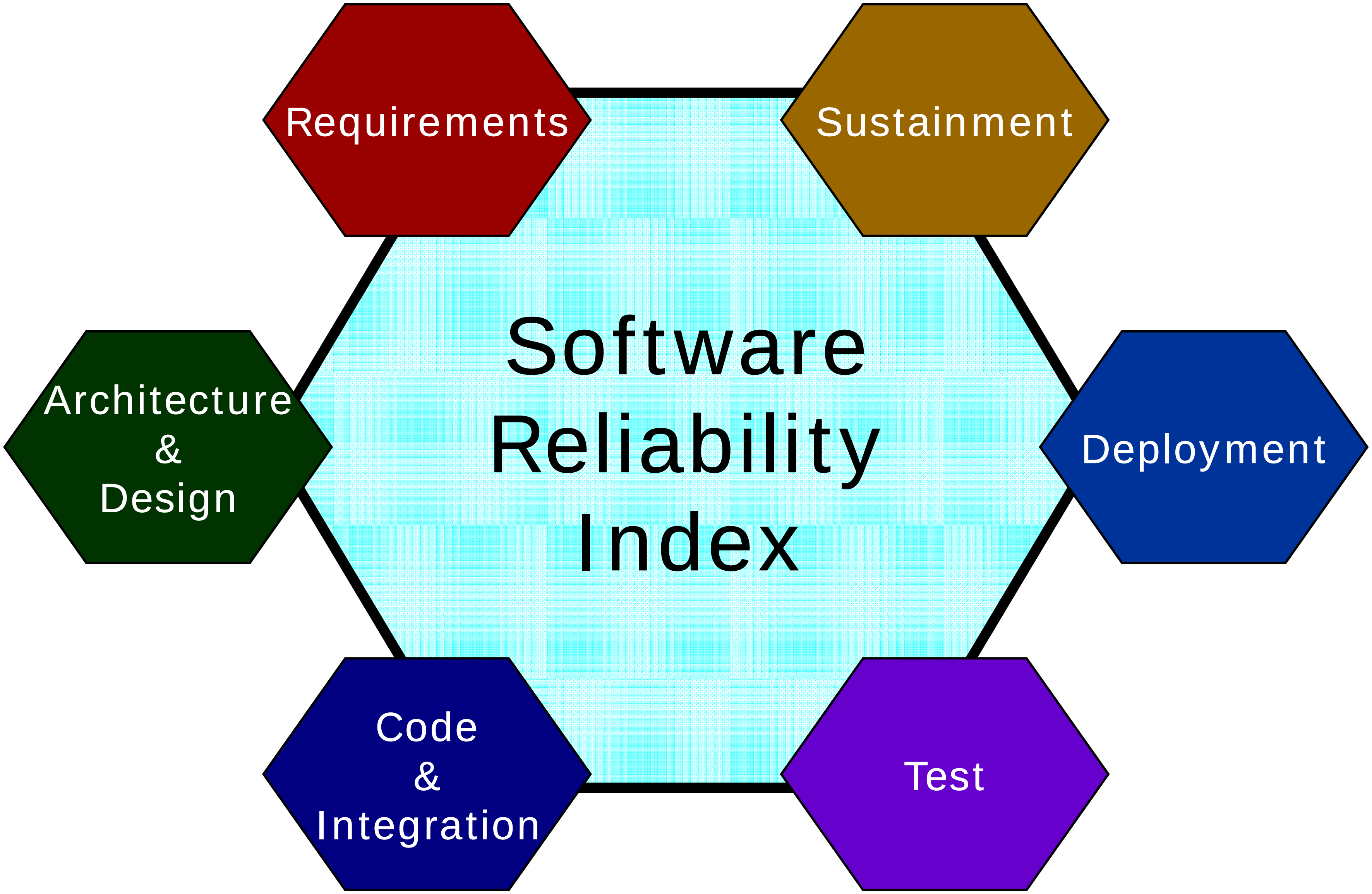
Software Lifecycle
Using Phases

- Advantages
  - While some has been developed for Software Reliability, much has been developed for each lifecycle phase
  - It is fairly easy to identify a few key reliability factors in each lifecycle phase
  - Each lifecycle phase could, in fact, have its own reliability index which would then feed into the overall index
- Disadvantages
  - While there is a lot of data for each lifecycle phase, the data have never been correlated to true reliability
  - Since there is little data on what constitutes reliability, it makes validating any reliability model difficult

SRI Parameters

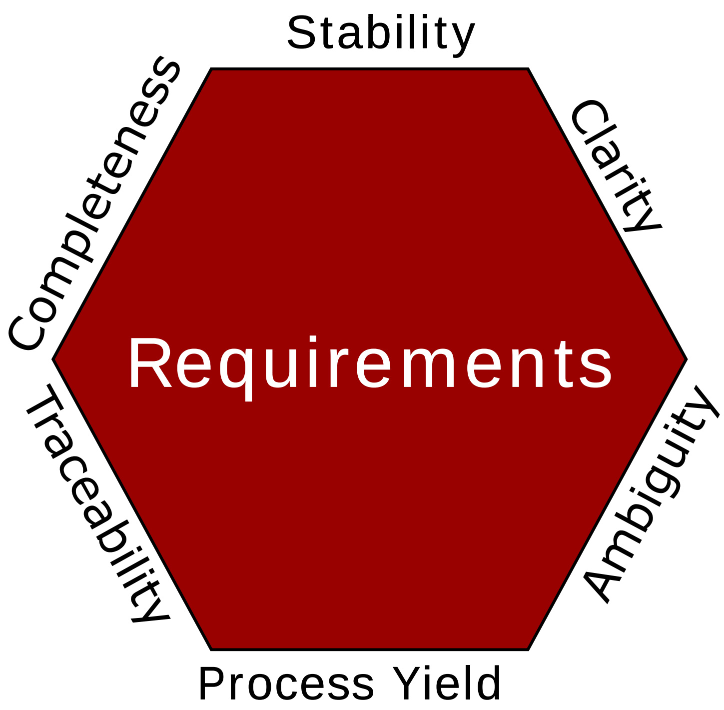


SRI Parameters



Software Reliability Index
Requirements

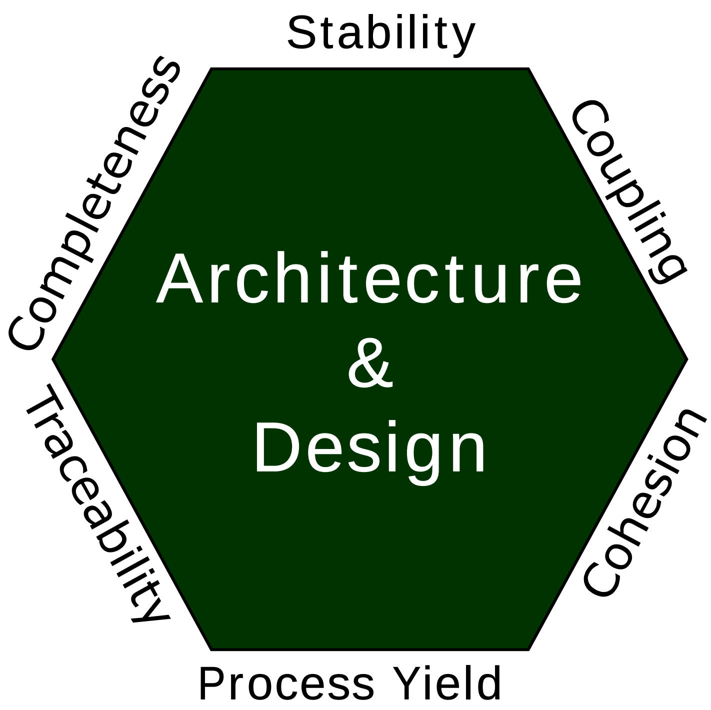
- **Stability**
  - **The percent of unchanged requirements per release**
  - **A new or deleted requirement is a changed requirement**
- **Clarity**
  - **The percent of requirements that are clear and understandable**
- **Completeness**
  - **The percent of requirements without TBDs, TBRs, TBAs, etc.**
- **Ambiguity**
  - **The percent of requirements with potential multiple meanings**
- **Traceability**
  - **The percent of requirements traced upward to a higher level document and traced to a lower level design component**
- **Process Yield**
  - **The percent of defects removed**



Software Reliability Index
Architecture & Design

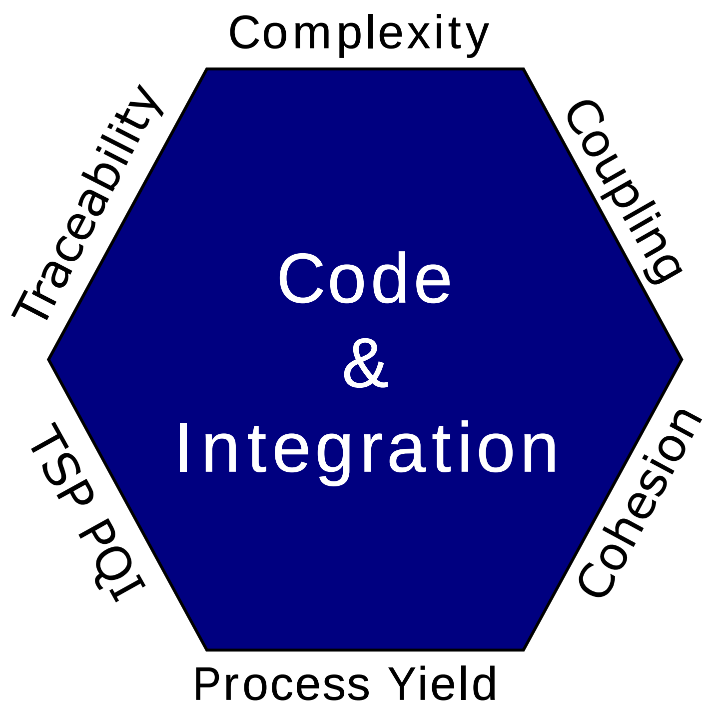
OGDEN AIR LOGISTICS CENTER

- **Stability**
  - **The percent of unchanged platform components**
  - **A new or deleted component is a changed component**
- **Interface Definition Completeness**
  - **The percent of completeness of Interface Control Documents**
- **Design Coupling**
  - **The percent of modules that exhibit low coupling**
- **Design Cohesion**
  - **The percent of modules that exhibit high cohesion**
- **Traceability**
  - **The percent of requirements traced upward to a higher level document and traced to a lower level design component**
- **Process Yield**
  - **The percent of defects removed**



Software Reliability Index
Code & Integration

- **Cyclomatic Complexity**
  - **McCabe Cyclomatic Complexity**
- **Code Coupling**
  - **The percent of modules that exhibit low coupling**
- **Code Cohesion**
  - **The percent of modules that exhibit high cohesion**
- **Traceability**
  - **The percent of requirements traced upward to a higher level document and traced to a lower level design component**
- **Process Quality Index (PQI)**
  - **The combined TSP PQI measure for all modules**
- **Process Yield**
  - **The percent of defects removed**

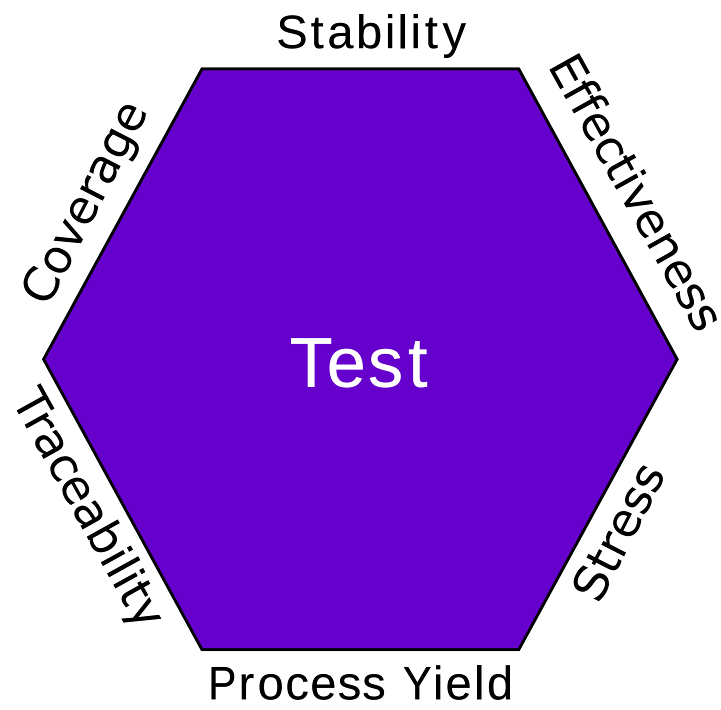


SRI_C

Software Reliability Index
Test

OGDEN AIR LOGISTICS CENTER

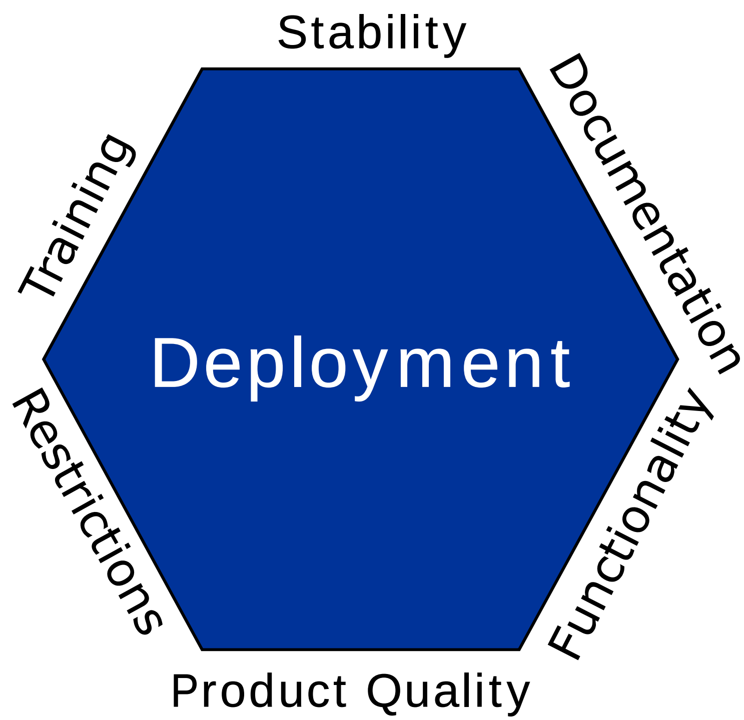
- **Coverage**
  - **The percent of requirements covered through testing**
- **Effectiveness**
  - **The level of confidence that existing defects are being found through testing**
- **Stress**
  - **The percentage of system components tested outside the expected limits**
- **Stability**
  - **The percent of requirements tested relative to the requirements implemented**
- **Traceability**
  - **The percent of requirements traced upward to a higher level document**
- **Process Yield**
  - **The percent of defects removed**



Software Reliability Index
Deployment

OGDEN AIR LOGISTICS CENTER

- **Training**
  - **The percentage of system features covered by training prior to deployment**
- **Documentation**
  - **The percentage of required documentation completed at time of deployment**
- **Stability**
  - **The percentage of the implemented system that is fully configured and supported at deployment**
- **Functionality**
  - **The percent of the total system requirements implemented at time of deployment**
- **Restrictions**
  - **The percent of implemented system requirements that are fully functional**
- **Product Quality**
  - **The percent of total defects found prior to deployment**

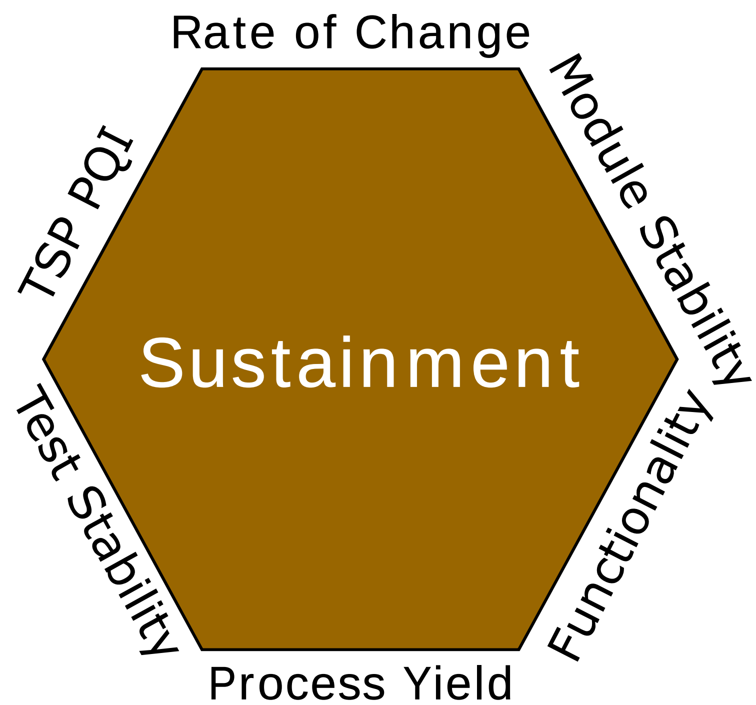


SRI_D

Software Reliability Index
Sustainment

OGDEN AIR LOGISTICS CENTER

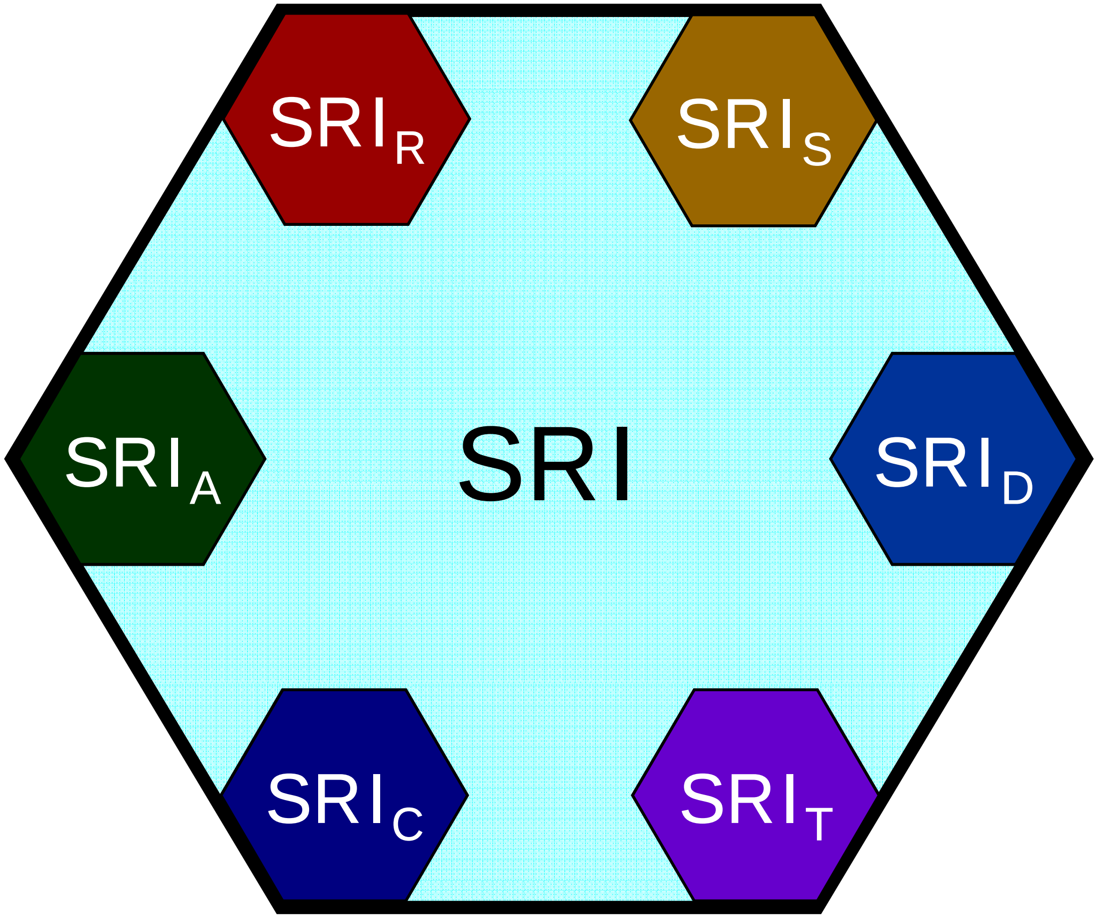
- **Rate of Change**
  - Percent of requirements unchanged per month
- **Module Stability**
  - Percent of total software modules unchanged
- **TSP Process Quality Index**
  - The combined TSP PQI measure for all updated modules
- **Test Stability**
  - The percent of requirements tested relative to the requirements implemented
- **Functionality**
  - The percent of functional improvement over the original release
- **Process Yield**
  - The percent of defects removed



SRI_S

Software Reliability Index

OGDEN AIR LOGISTICS CENTER



$$SRI = SRI_R * SRI_A * SRI_C * SRI_T * SRI_D * SRI_S$$

Pros and Cons

- Pros
  - Each lifecycle phase can be measured independently or in concert with the others
  - Data on each lifecycle phase are derived from data which are typically collected in these phases
  - Collection and examination of these data encourages a high maturity approach to the software development life cycle, which has been proven to produce reliable software
- Cons
  - 36 is a large number of factors, many of which may not be controllable
  - Statistical analysis of these factors has not been conducted to determine the relevance of each to SRI
  - The index may be unstable, since the SRI is a product of products and a major variation of any one factor or minor variations of several factors can have a major influence on the outcome

Conclusions

- Software does not wear out like Hardware, but its Reliability is an important contributor to overall System Reliability
- Most measures for Software Reliability concentrate on the Development or Testing phases
- Full-lifecycle (i.e. Requirements phase through to Sustainment phase) reliability measures potentially provide a more comprehensive assessment
- An overall Software Reliability Index (SRI) can be computed as a product of Reliability Indices from each lifecycle phase
- The SRI can be used to track reliability over time, used as a predictor to compare with actual reliability, or to identify areas of improvement that will increase reliability
- The SRI model presented here is just now being put into practice for EFV software and evidence of its efficacy will be published at a later date

Questions???

Contact Information

OGDEN AIR LOGISTICS CENTER

- The MITRE Corporation / PM AAA, USMC
  Robert J. Knapper
  (703) 496-9237
  rknapper@mitre.org
  knapperrj.ctr@efv.usmc.mil

- 309th Software Maintenance Group/
  520th Software Sustainment Squadron
  David R. Webb
  (801) 586-9330
  david.webb@hill.af.mil

  David L. Jolley
  (801) 777-3823
  david.jolley@hill.af.mil